Automatic Data Generation for Web Application Validation.

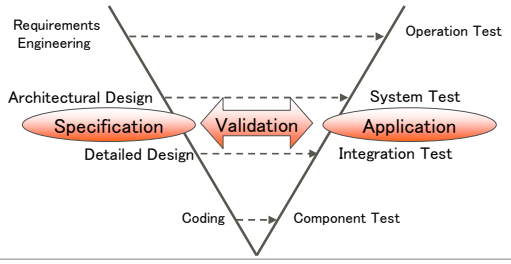
Shoichiro Fujiwara,
Software Innovation Laboratory,
Fujitsu Laboratories.

Outline

- 1.Background and motivation for automatic web application validation
- 2.SWEEP: a validation tool for server-side Java applications
- 3.Driver/stub data generation for SWEEP
- 4.Concluding remarks

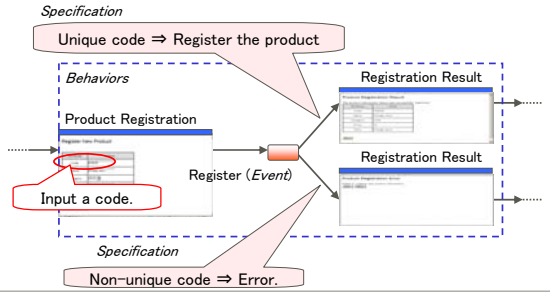
Target Domain & Phase

- Domain: Web applications (server-side Java)
- Phase: Integration or system test

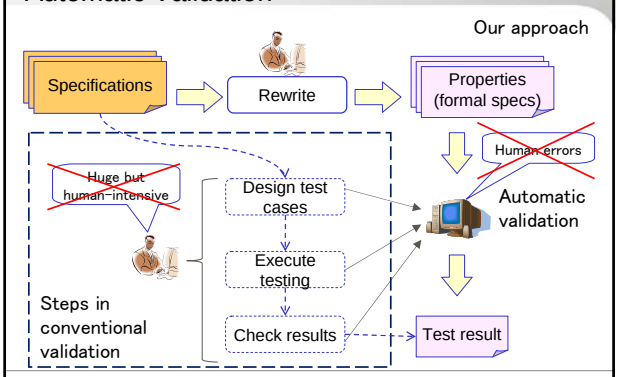


Web App Validation

- Checking whether application *behaviors* around each *event* fulfill the *specifications*.



Motivation and Approach toward Automatic Validation

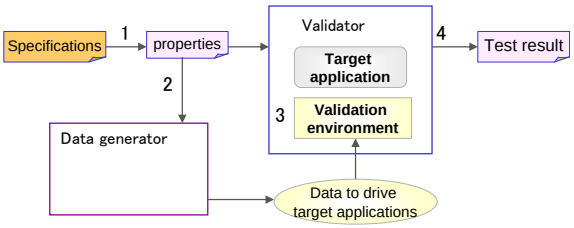


Outline

- 1.Background and motivation for automatic web application validation
- *2.SWEEP: a validation tool for server-side Java applications*
- 3.Driver/stub data generation for SWEEP
- 4.Concluding remarks

Validation Steps

1. Rewrite specifications into properties.
2. Generate data for the target application to drive.
3. Construct a validation environment.
4. Validate the target.



Rewriting Specifications into Properties

Specifications

Business Function: Product Registration

When the button "Register" is pressed,	• *If code is not unique, then display an error message "PR001E".* • If price is less than 0, then display an error message "PR002E". • Otherwise, register the product information.
Invariant	*Each product's price in DB must not be less than 0.*

Rewrite manually

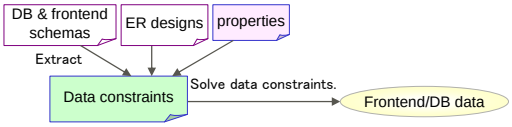
Properties

Described as "Pre-condition → Post-condition"

No.	Event	Pre-condition	Post Condition
1	Register	∃rec in ProductTable s.t. rec.code = input.code	output.message = "PR001E"
2	-	-	∀rec in ProductTable, rec.price >=0
...	...	...	...

Generate Data for the Target Application

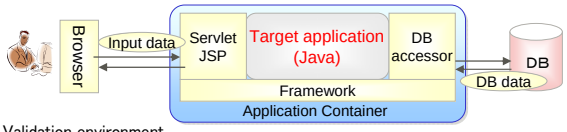
- Extract constraints from design information in addition to properties.
  - Schemas of DB & frontends, ER designs like cardinality
- Solve the constraints to generate frontend/DB data.



Validation Environment Construction

- Utilize framework, design and properties.

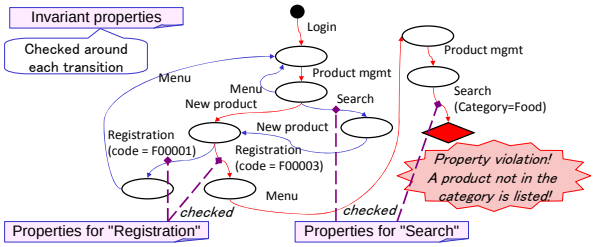
Web application in actual use



Validation environment
(run on the validator.)

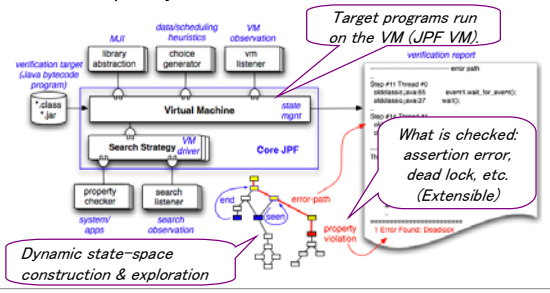
Validation

- Exploring program states and checking properties.
  - Program state: frontend and DB states
  - Engine: JPF (Java PathFinder) with extensions



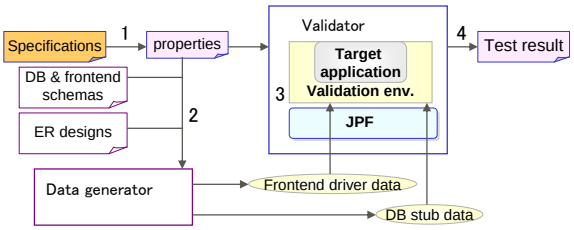
Java PathFinder

- A model checker for Java byte code
  - Developed by NASA Ames Research Center



Validation Steps Again

1.Rewrite specifications into properties.
2.Generate data for a frontend driver and a DB stub.
3.Construct a validation environment.
4.Validate the target.

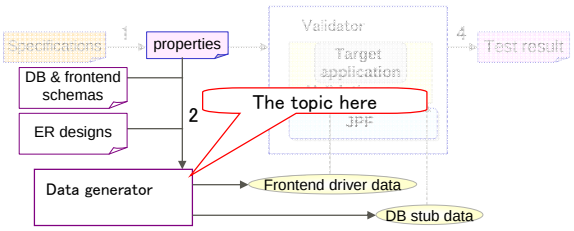


Outline

- 1.Background and motivation for automatic web application validation
- 2.SWEEP: a validation tool for server-side Java applications
- *3.Driver/stub data generation for SWEEP*
- 4.Concluding remarks

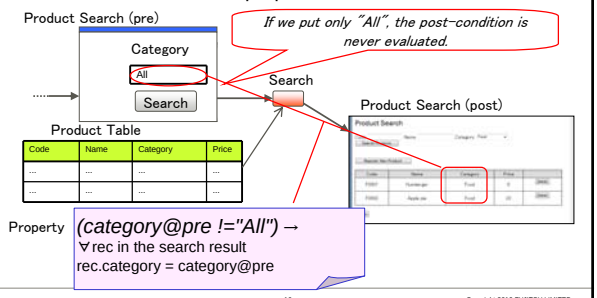
Validation Steps Again

1.Rewrite specifications into properties.
2.Generate data for a frontend driver and a DB stub.
3.Construct a validation environment.
4.Validate the target.



Data: Key for "Good" Validation

- Need *a corresponding data set for each property*.
  - Data decide whether properties are checked or not.



Data Constraints

- What we need to consider when generating data.
  - *Pre-condition of properties*
  - *Invariant properties*
    - Considered to be a kind of pre-conditions
  - *Type constraints*
    - String/integer, length, allowed characters, etc.
  - *Table constraints*
    - Foreign key relations, cardinality, uniqueness, etc.

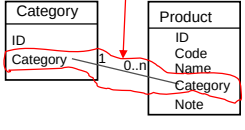
Data Constraint Example in Design

(DB) Product table schema

Column	Key	Unique	Type	length	Possible chars
ID	key	unique	Integer	10	-
Code	-	unique	String	10	_a-zA-Z0-9
Name	-	-	String	40	_a-zA-Z0-9
Category	-	-	String	40	_a-zA-Z0-9
Note	-	-	String	1000	Any

Table constraints

(ER constarints)
ER diagram



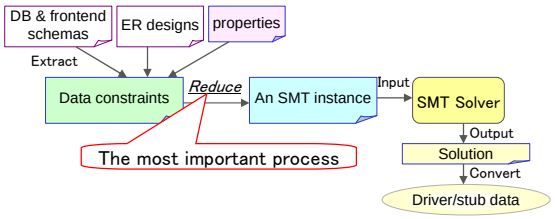
Type constraints

(Frontend) Product registration schema

Column	Type	length	Possible chars
ID	Integer	10	-
Code	String	10	_a-zA-Z0-9
...	...	...	...

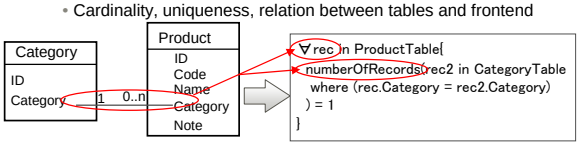
Automatic Data Generation Architecture

- Use an SMT (Satisfiability Modulo Theories) solver.
  - SMT: satisfiability problem of some first order formulae
  - SMT solvers: Yices (SRI), CVC3 (NYU and U Iowa), Z3 (Microsoft Research), etc.



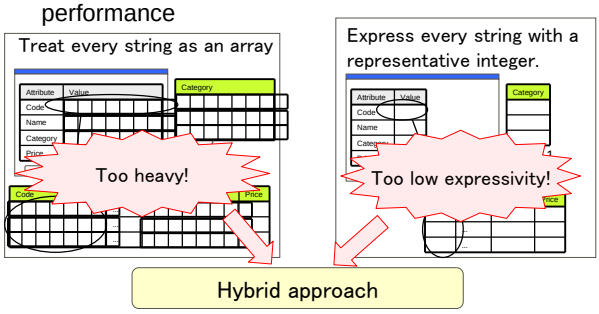
Problems in Data Constraint Reduction

- *String encoding*
  - The tool must decide many strings in a practical time.
    - A Web application generally handles much string data.
- *Reduction of table size constraints*
  - Constraints on the number of records are common and generally undecidable.
    - Cardinality, uniqueness, relation between tables and frontend



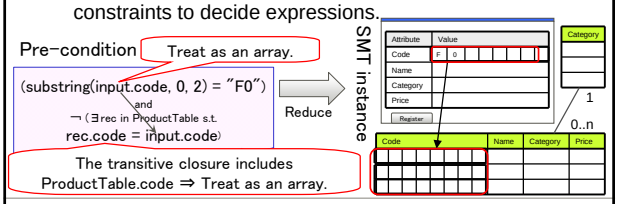
String Encoding (1)

- Trade off between abstraction level and performance



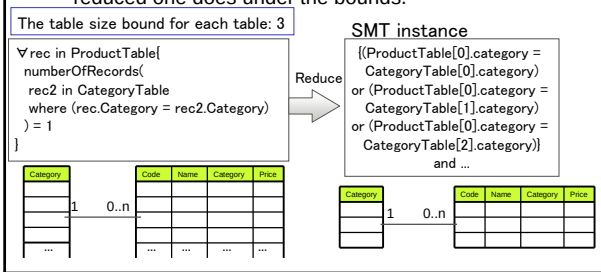
String Encoding (2)

- Hybrid approach
  - Strings free from complex constraints like substring ⇒ expressed with representative integer variables.
  - The others ⇒ expressed as (numeric) arrays.
  - Analyze transitive closure of binary relations from data constraints to decide expressions.



Structured Constraint Reduction

- Bound table sizes
  - Reduce a constraint so that the original one holds iff the reduced one does under the bounds.



Convert Solutions into Driver/Stub Data

- Assign string values to representative integer values preserving order relations.
- We can give strings to a transitive closure without taking care of the other closures.

SMT solution

Attribute	Value
Code	70 48 48 48 49 0
Name	1000
Category	1100
Price	20

Convert

Driver/Stub data

Attribute	Value
Code	F0001
Name	Product3
Category	Food
Price	20

Code	Name	Category	Price
70 48 48 48 49 0	2000	1100	10
70 48 48 48 50 0	2100	1100	15

Code	Name	Category	Price
F0001	Product1	Food	10
F0002	Product2	Food	15

Trial with a Real Project Asset

FUJITSU

- What we did
  - Data generation for 23 properties, which specify the web application behavior around an event.
    - 13 normal cases and 10 error cases
    - Magnitude of data : 10 DB tables, half of which are master tables
- Result
  - The application ran successfully with the generated data.
  - Each property was evaluated at least once during validation.
  - It took about 5 minutes to generate data from each property.

Outline

FUJITSU

- 1.Background and motivation for automatic web application validation
- 2.SWEEP: a validation tool for server-side Java applications
- 3.Driver/stub data generation for SWEEP
- *4.Concluding remarks*

Concluding Remarks

FUJITSU

- Validating web application by SWEEP
  - Rewrite specifications into properties.
  - Stub/driver data generation from the properties.
  - Validation environment with stubs/drivers.
  - Automatic validation based on JPF.
- Automatic stub/driver data generation
  - Extract data constraints from the properties and other design information.
  - Reduce the constraints into an SMT instance.
  - Solve the instance with an SMT solver.
  - Convert the solution into the stub/driver data.

FUJITSU

shaping tomorrow with you